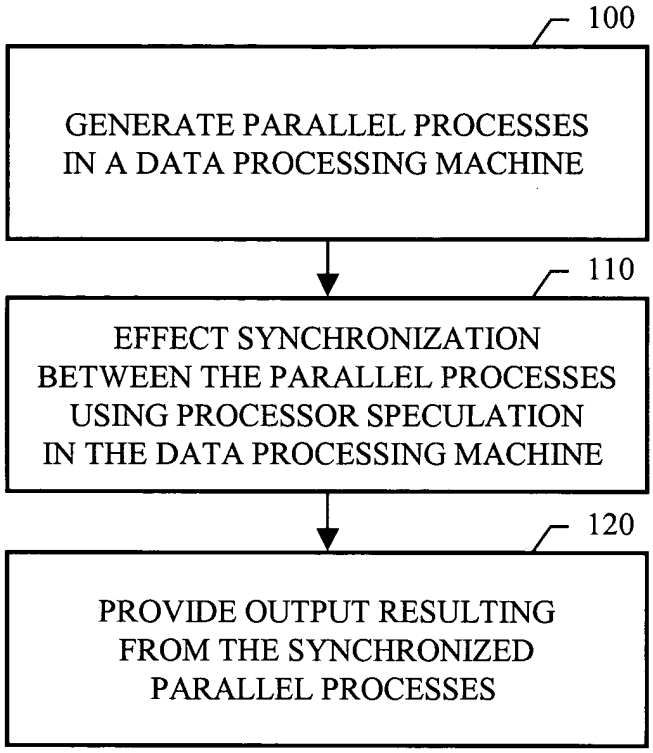
100
GENERATE PARALLEL PROCESSES IN A DATA PROCESSING MACHINE
110
EFFECT SYNCHRONIZATION BETWEEN THE PARALLEL PROCESSES USING PROCESSOR SPECULATION IN THE DATA PROCESSING MACHINE
120
PROVIDE OUTPUT RESULTING FROM THE SYNCHRONIZED PARALLEL PROCESSES

FIG. 1

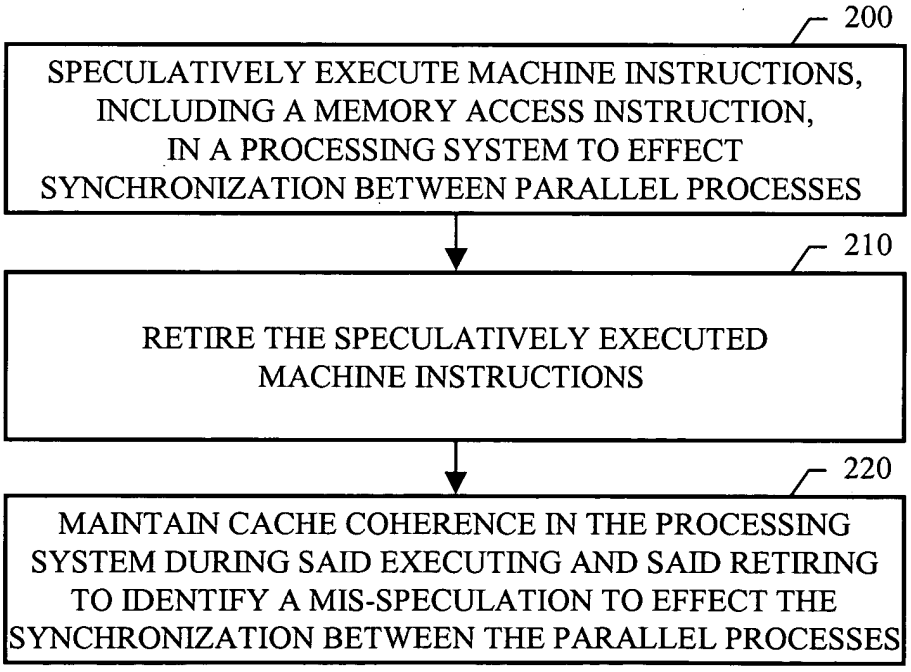
200
SPECULATIVELY EXECUTE MACHINE INSTRUCTIONS, INCLUDING A MEMORY ACCESS INSTRUCTION, IN A PROCESSING SYSTEM TO EFFECT SYNCHRONIZATION BETWEEN PARALLEL PROCESSES
210
RETIRE THE SPECULATIVELY EXECUTED MACHINE INSTRUCTIONS
220
MAINTAIN CACHE COHERENCE IN THE PROCESSING SYSTEM DURING SAID EXECUTING AND SAID RETIRING TO IDENTIFY A MIS-SPECULATION TO EFFECT THE SYNCHRONIZATION BETWEEN THE PARALLEL PROCESSES

FIG. 2

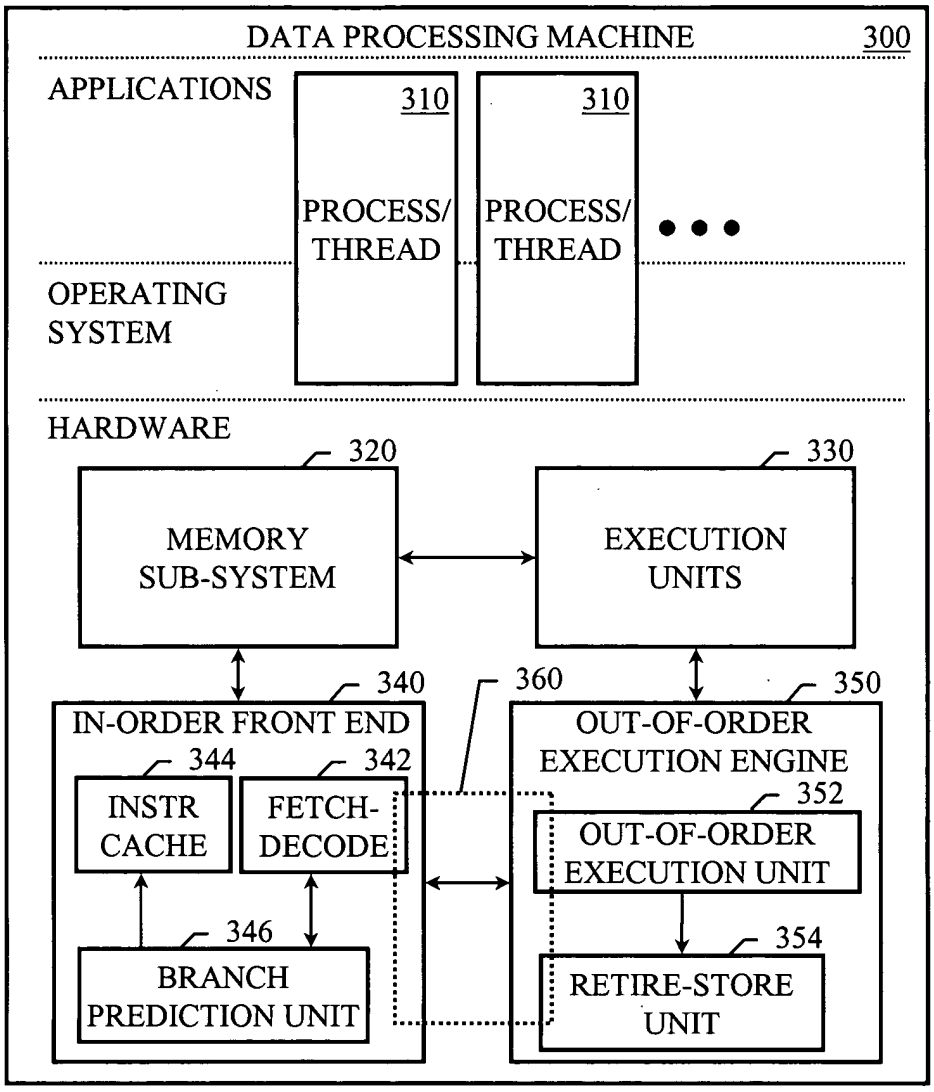
DATA PROCESSING MACHINE
300
APPLICATIONS
310
PROCESS/ THREAD
310
PROCESS/ THREAD
OPERATING SYSTEM
HARDWARE
320
MEMORY SUB-SYSTEM
330
EXECUTION UNITS
340
IN-ORDER FRONT END
344
INSTR CACHE
342
FETCH-DECODE
346
BRANCH PREDICTION UNIT
360
350
OUT-OF-ORDER EXECUTION ENGINE
352
OUT-OF-ORDER EXECUTION UNIT
354
RETIRE-STORE UNIT

FIG. 3

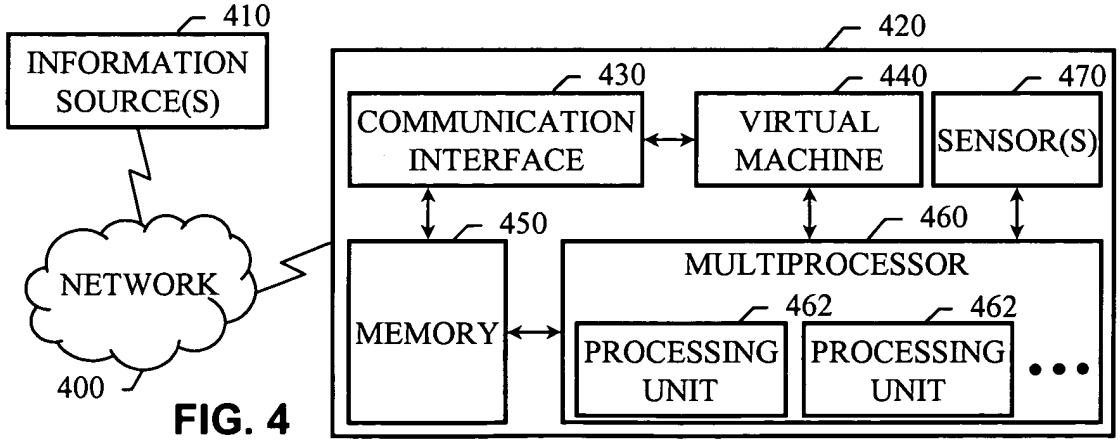
410
INFORMATION SOURCE(S)
NETWORK
400
420
430
COMMUNICATION INTERFACE
440
VIRTUAL MACHINE
470
SENSOR(S)
450
MEMORY
460
MULTIPROCESSOR
462
PROCESSING UNIT
462
PROCESSING UNIT

FIG. 4

```
volatile int lock_var;          // lock_var is the lock variable
 if(lock_var==0){               // this denotes unlocked state
   loc=0;
   spec loc,shoot_down;         //begin speculation, goto shoot_down if
                                //  misspeculated
   rl=lock_var;                 //line 1
   if(rl==0){                   //line 2
     lock_var=tid;              //line 3
   }
   commit loc,shoot_down;       //start retiring, goto shootdown on violation

   if(lock_var==tid){           //if true then got the lock
     CS;                        //critical section
     lock_var=0;                //unlock
     goto post_lock;
   }

shoot_down:                     //there was a conflict, do the usual atomic
                                //  operation
   grab lock the conventional way
   ...

post_lock:
   normal program flow
   ...
```

500

FIG. 5

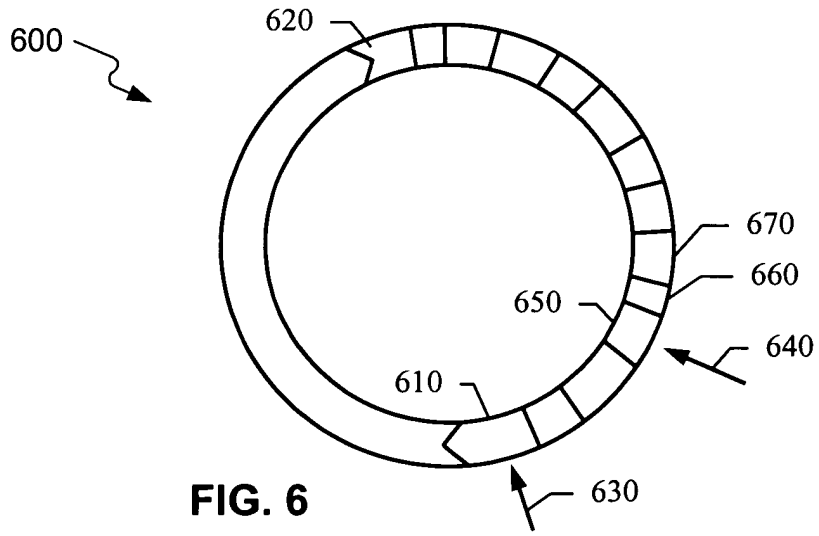

FIG. 6